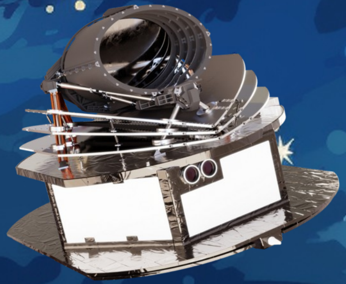
Signal collection and noise in the Ariel era

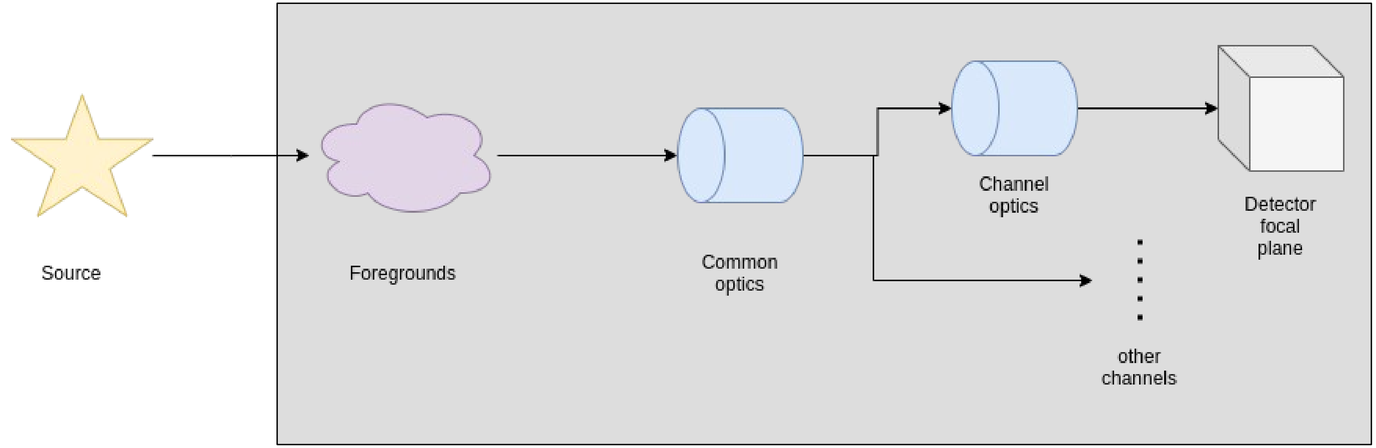
Lorenzo V. Mugnai

(MugnaiL@cardiff.ac.uk)



Pic. credits Lorenzo V Mugnai

Road to focal plane



This presentation will briefly overview the notions needed to define the best strategy to **observe**, **collect and reduce** data.

1)Understand the light source;
2)Understand the collecting instrument;
3)Understand the detector.

Source light to the focal plane

- We start with the source Spectral Energy Density,

$$S(\lambda) = \frac{R_\star^2}{D^2} S_\star(\lambda)$$

- Has unit of [W/(m^2 µm)]
- We need to obtain the Flux in [counts/s]

$$F(x_i, y_i) = ??S(\lambda)$$

- We consider slit-less spectrometers

Source light to the focal plane

First we remove the dependency from the surface, by multiplying for the telescope aperture

$$F(x_i, y_i) \approx A_{tel} S(\lambda)$$

We now moved from [W/(m^2 µm)] to [W/µm]

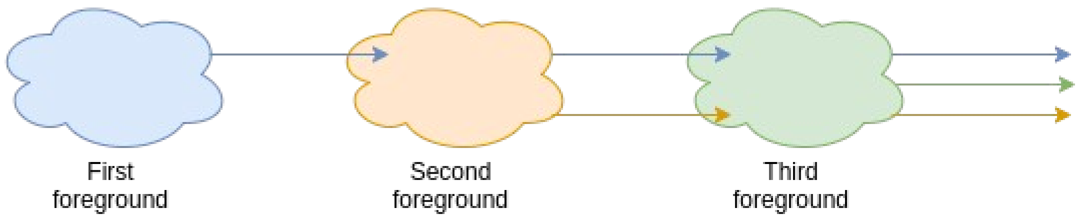
Source light to the focal plane

In a photometer, each pixel receive light from the full wavelength range.
In a spectrometer, each pixel receive light only from a certain wavelength range.
In general, the pixel collect all the wavelength that hit it.

$$F(x_i, y_i) \approx A_{tel} \int_{\lambda} S(\lambda) d\lambda$$

However, not all the wavelengths reach the focal plane…

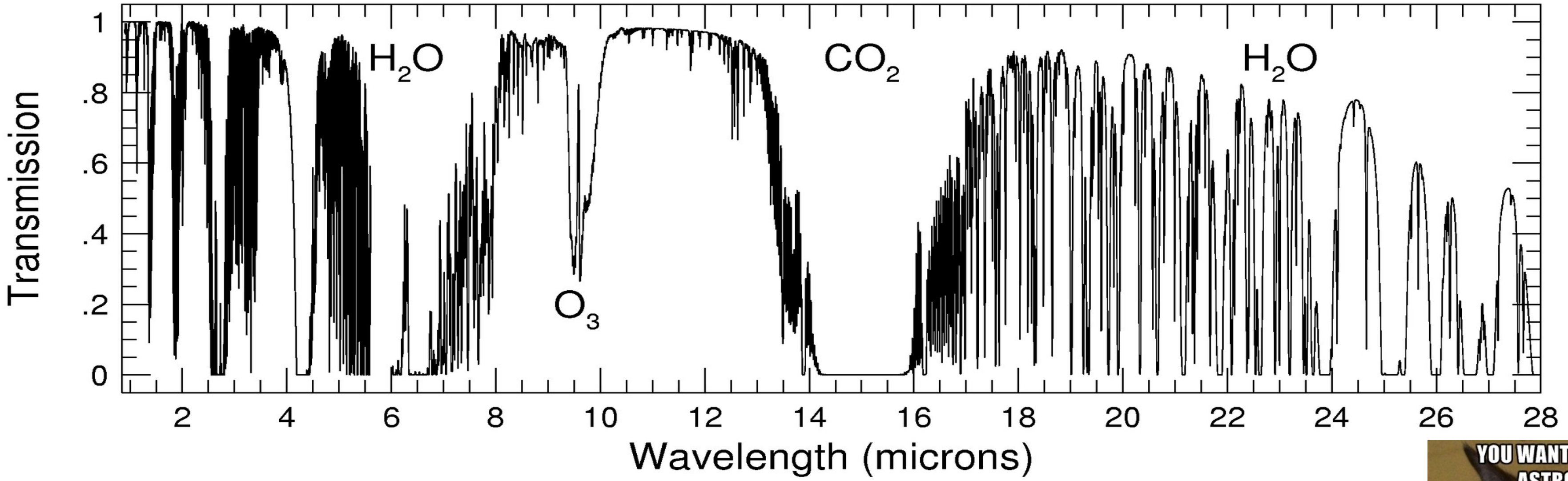
Foregrounds



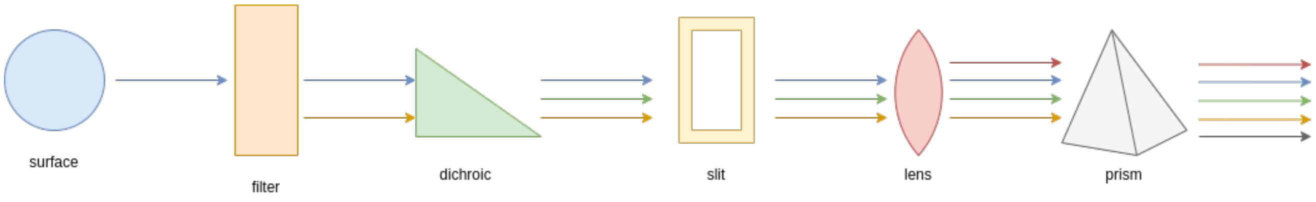
Foregrounds are “filters” between the source and the telescope.

Each foreground absorbs part the incoming light and emits light itself.

Example: Earth atmosphere



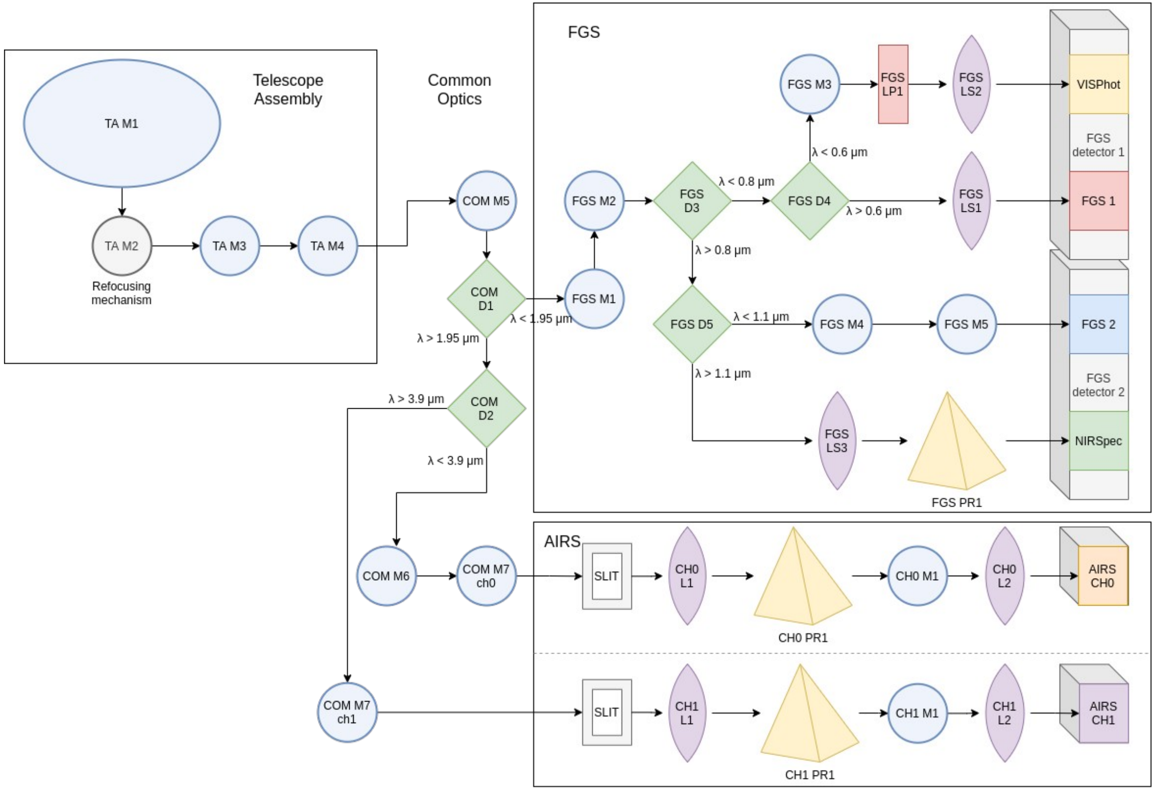
Optical path



Like the foregrounds, optical elements are “filters” between the source and the telescope.

Each element absorbs part the incoming light and emits light itself.

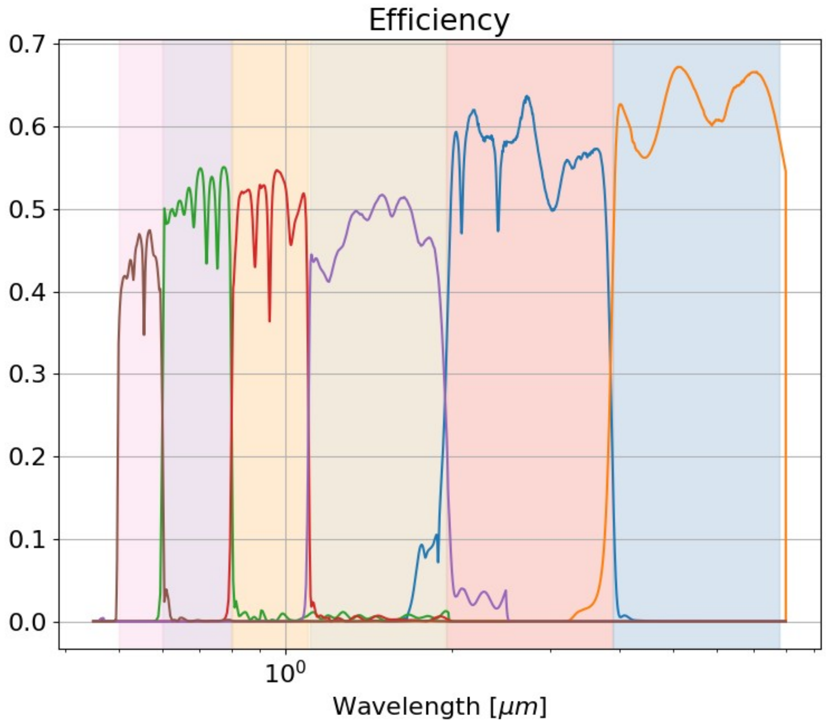
Ariel Optical Path



Total transmission

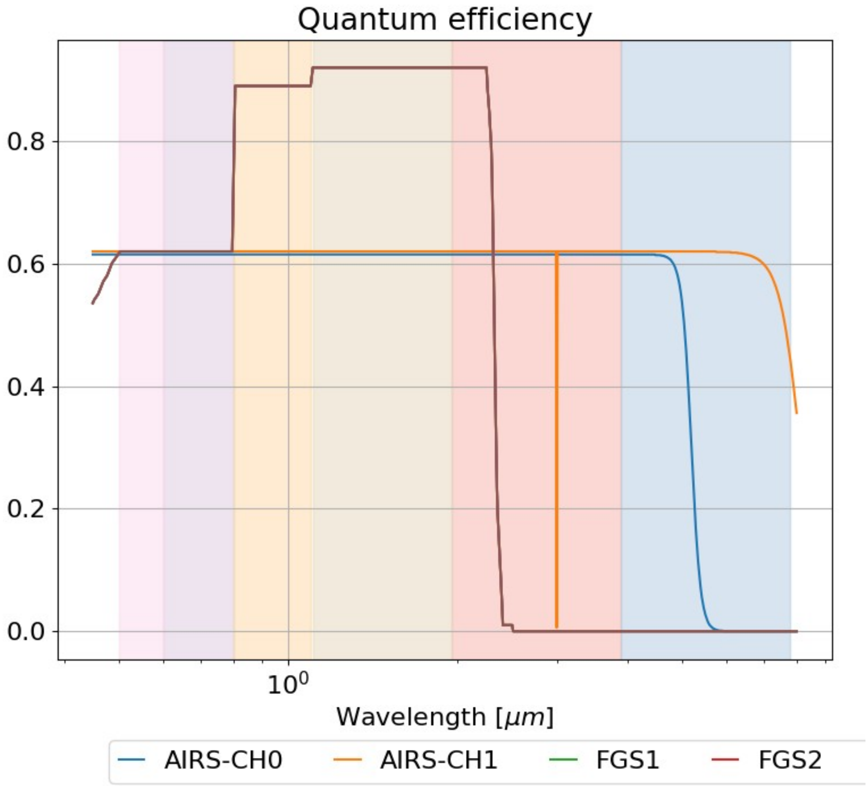
The product of all the optical elements transmissions is the **optical transmission efficiency**.

If foreground are included, they contribute to the total transmission.



Pixel QE

Detector are not sensitive to every wavelength.



Source light to the focal plane

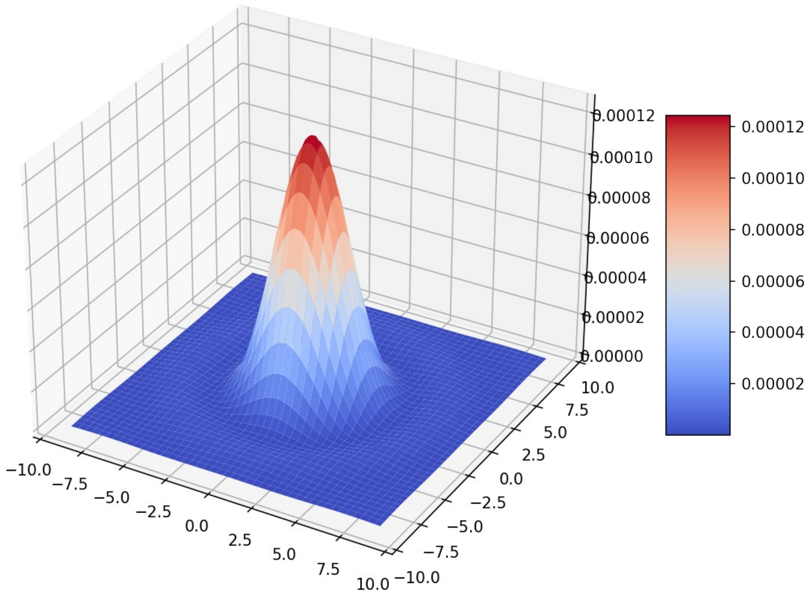
In general, the pixel collect all the wavelength that hit it, and convert the light according to its quantum efficiency

$$F(x_i, y_i) \approx A_{tel} \int_{\lambda} \Phi(\lambda)\, \nu(\lambda)\, S(\lambda) \frac{\lambda}{hc} d\lambda$$

ϕ Optical element transmission
ν Detector QE

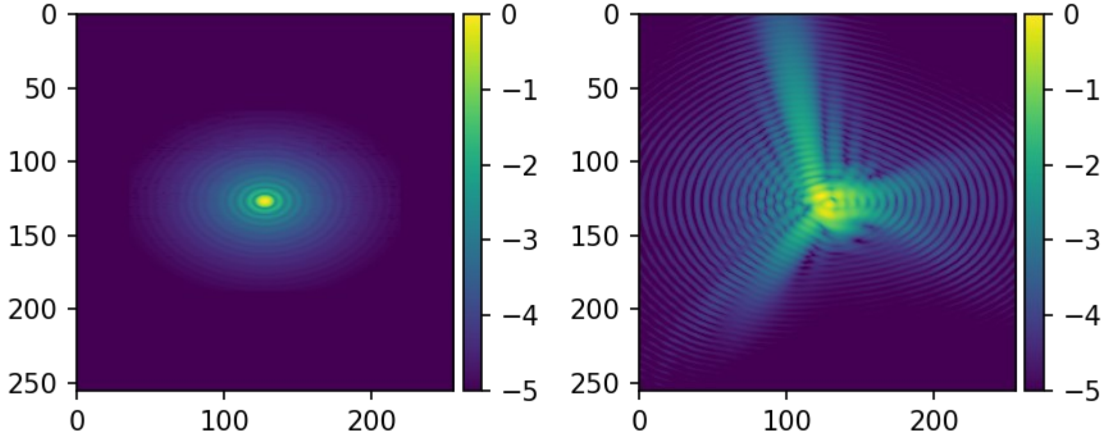
$\frac{\lambda}{hc}$ allows us to move from [W] to [count/s]

Point Spread Function



A point source in the sky generate a response on the focal plane called Point Spread Function (PSF).

The PSF can be affected by aberration.

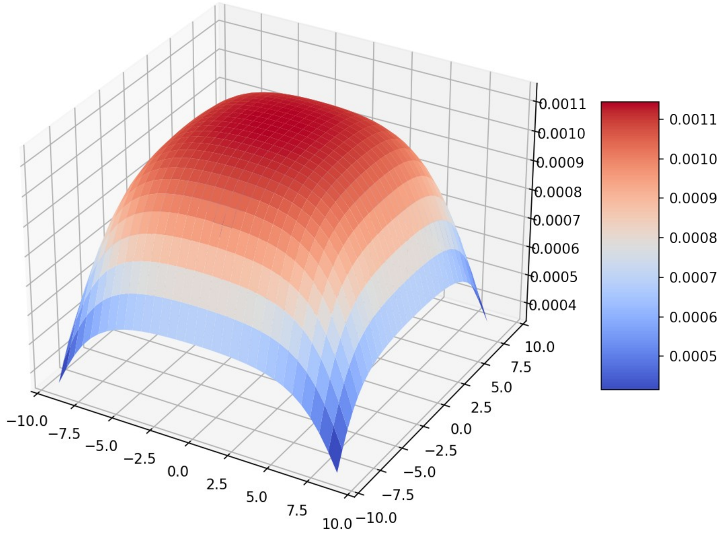


Point Response Function

Pixel response is not the same in all the surface

We need to combine the effect of the PSF and pixel in the Point Response Function

$$PRF(x_l, y_m, \lambda) = \int_u \int_v PSF(u, v, \lambda) H_{pix}(x_l - u, y_m - v) \, du \, dv$$



Source light to the focal plane

The incoming signal needs to be convolved with the PRF

$$F(x_i, y_i) = A_{tel} \int_u \int_v \int_\lambda \Phi(\lambda)\, \nu(\lambda)\, PRF(x_i - u, y_i - v, \lambda)\, S(\lambda) \frac{\lambda}{hc} d\lambda\, du\, dv$$

ϕ Optical element transmission
ν Detector QE

What is noise?

- Any photon reaching the focal plane that is not from the target source
- Any electron generated in the detector that is not generated by the target source
- Any variability on the signal not induced directly from the source

$$F(x_i, y_i) = A_{tel} \int_u \int_v \int_\lambda \Phi(\lambda)\, \nu(\lambda)\, PRF(x_i - u, y_i - v, \lambda)\, S(\lambda) \frac{\lambda}{hc} d\lambda\, du\, dv$$

$\Phi(\lambda, T)$ $\nu(\lambda, T, t)$ $PRF(x, y, \lambda, T, z)$

Other sources of photon?

We finally have the source flux on the focal plane…

…but what about foregrounds and optical elements?

Diffuse light

Emission from foregrounds and instrument produce a diffuse light background

$$F_o = A_{tel}\,\Omega \int_{\lambda} \prod_{j=o+1} \phi_j(\lambda)\,\nu(\lambda)\,I_o(\lambda)\,\frac{\lambda}{hc}\,d\lambda$$

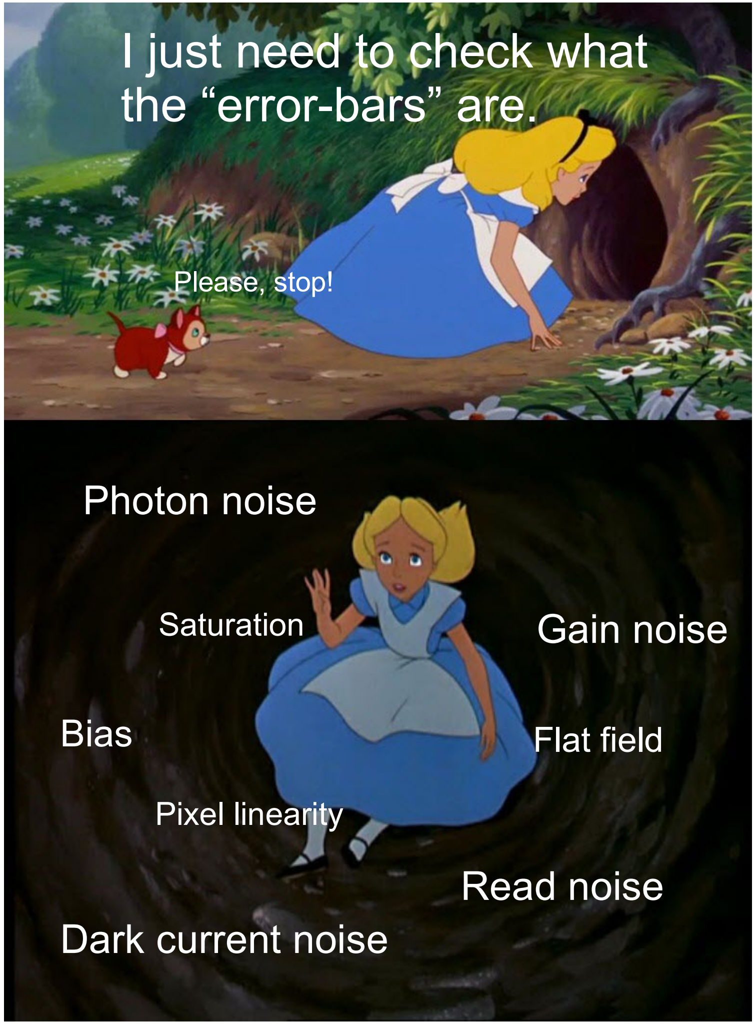
Ω Pixel solid angle
ϕ Optical element transmission
ν Detector QE
I_o Optical element emission

What about other noise?

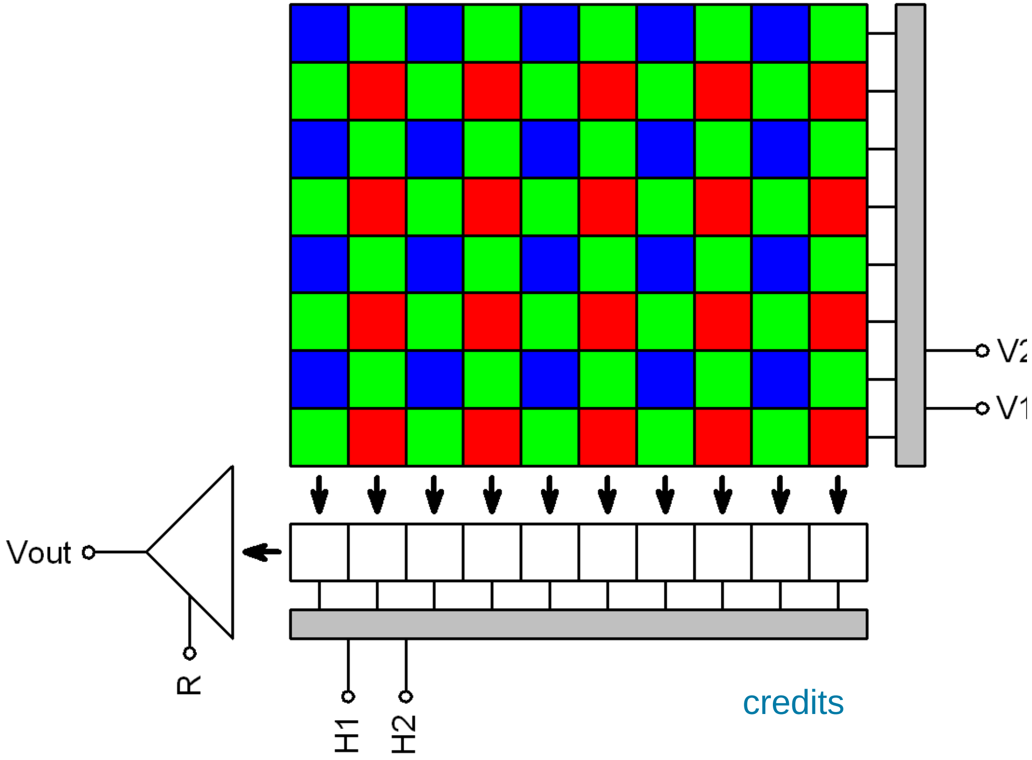
So far we estimated the counts generated by a pixel for an incoming light
… but **where is the noise coming from**?

- What is a pixel?
- How is the light collected?
- How are they read?
- Is it electronics involved?

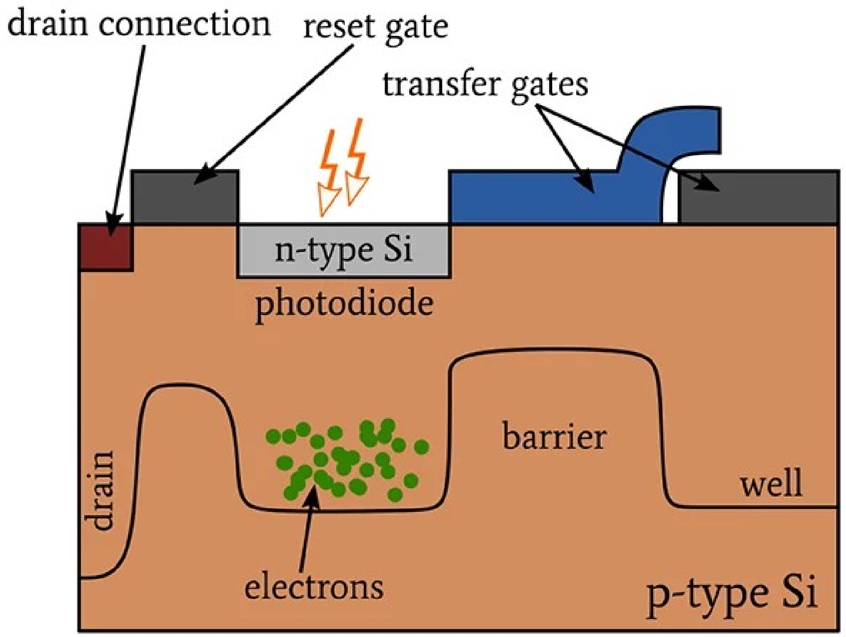
We are transforming a piece of metal in an eye: it is kinda of magic!



What's a detector?

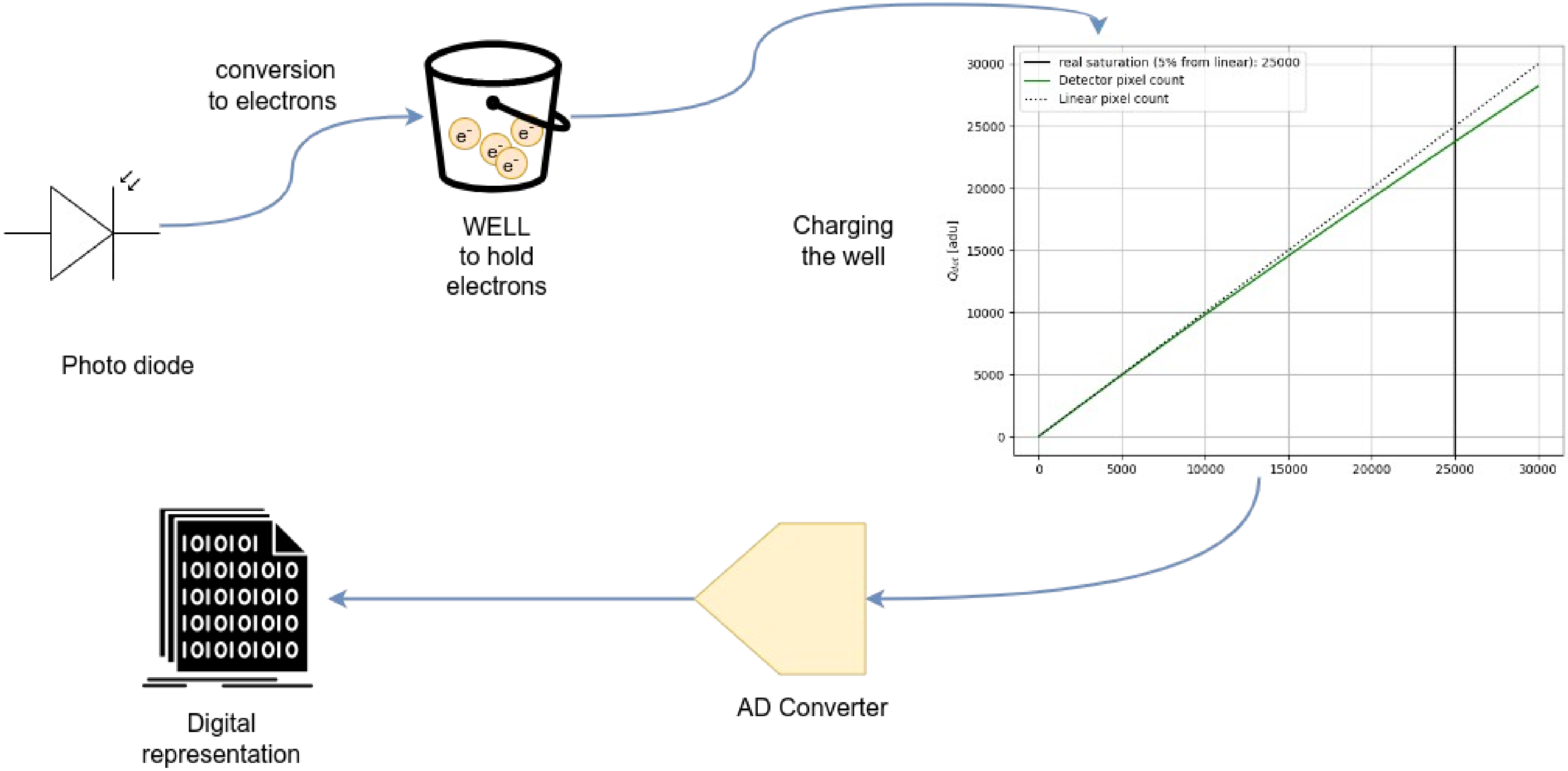


credits



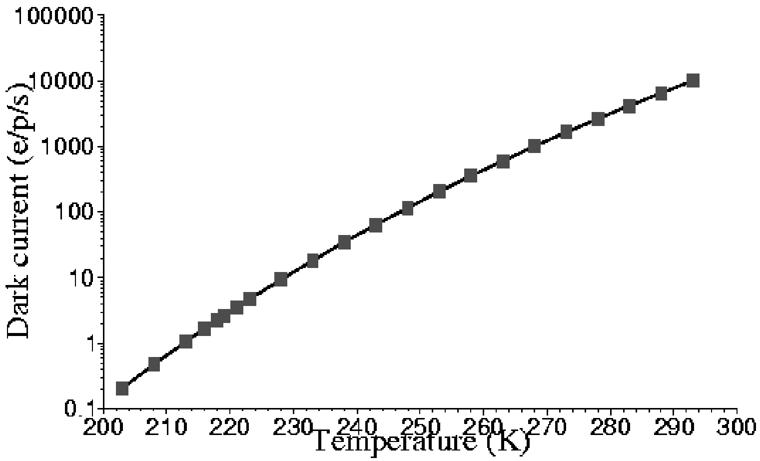
credits

How a sensor works

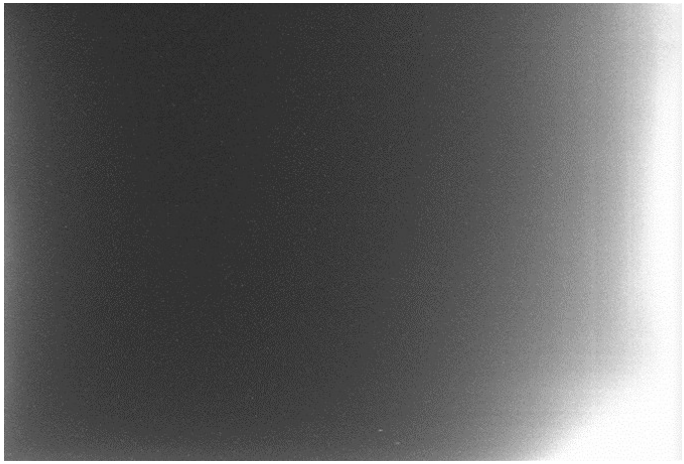


Dark Current

- It indicates how many electrons build up on each pixel for every second of exposure, typically shown as e–/p/s.
- It depends on the temperature



credits



credits

Reset Noise (kTC)

- Prior to the measurement, the detector is reset to a reference level.
- Noise is generated by an uncertainty in the reference voltage level due to thermal variations in the channel resistance of the reset transistor.

$$n_{kTC} = \frac{\sqrt{kTC}}{q}$$

Shot Noise

Because collecting light is a “counting” problem, it is subject to **Poisson statistic**. Poisson noise is called “shot noise”

$$\sigma_{shot} = \sqrt{N}$$

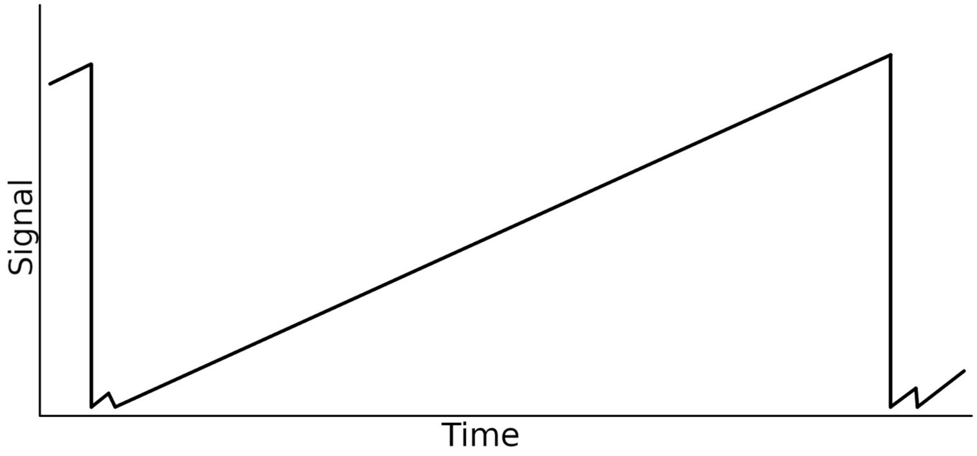
This is the **fundamental limit**: you can never get a better measurement than this.

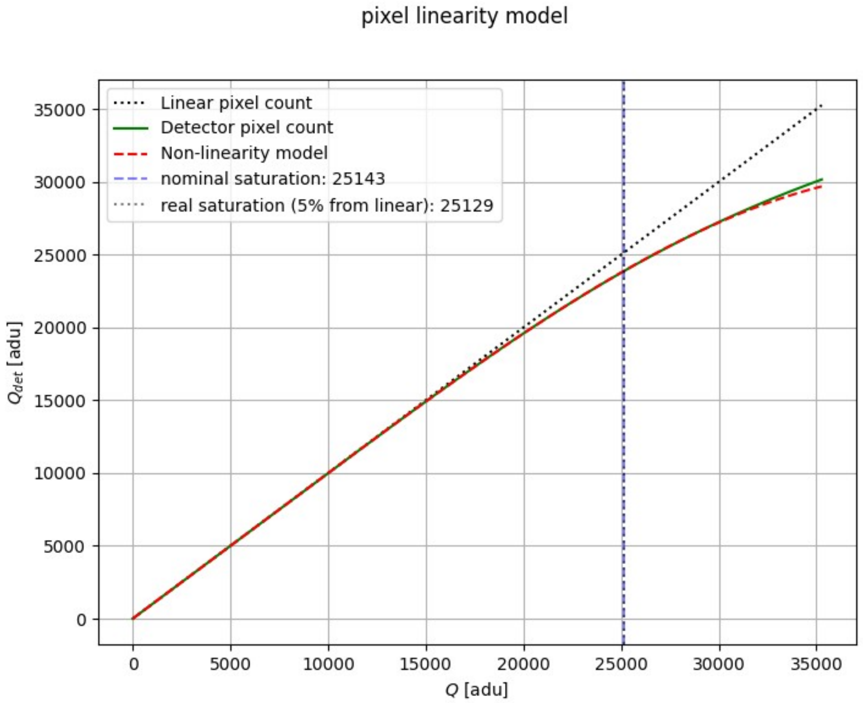
It is a random process: white noise.

Always compare your error-bars to the shot noise to see how far are you from ideal.

Pixel linearity

The well is finite. The pixel response depends on how filled the well is.



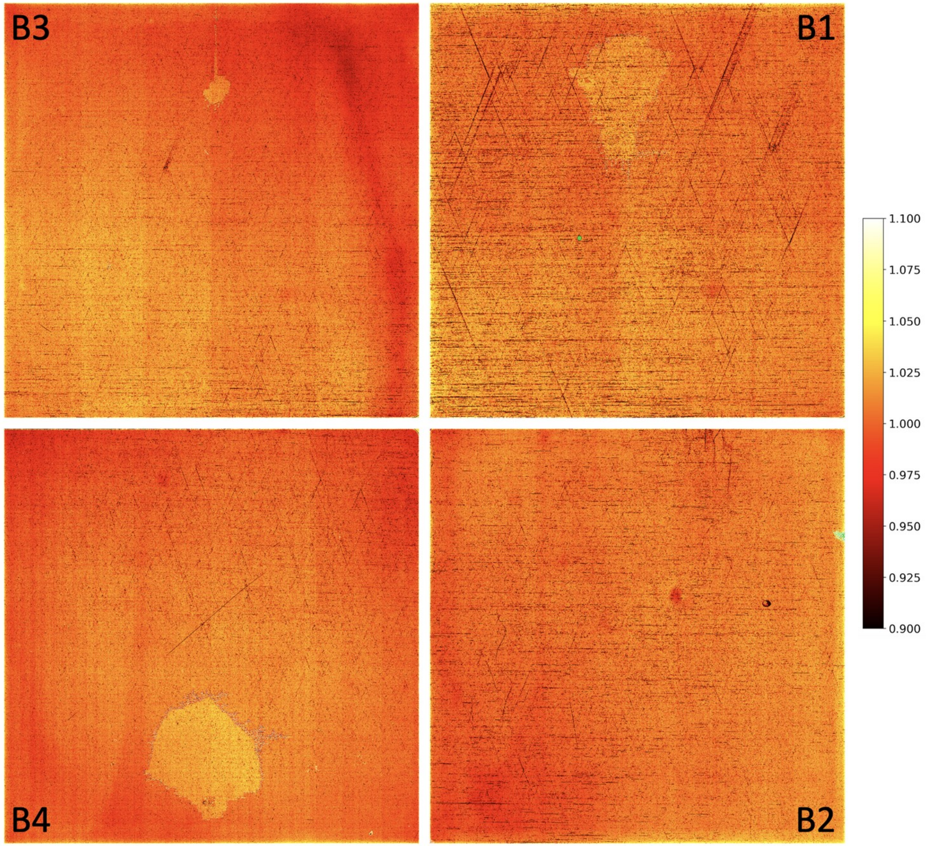


QE map

Pixel-to-pixel sensitivity changes across the detector.

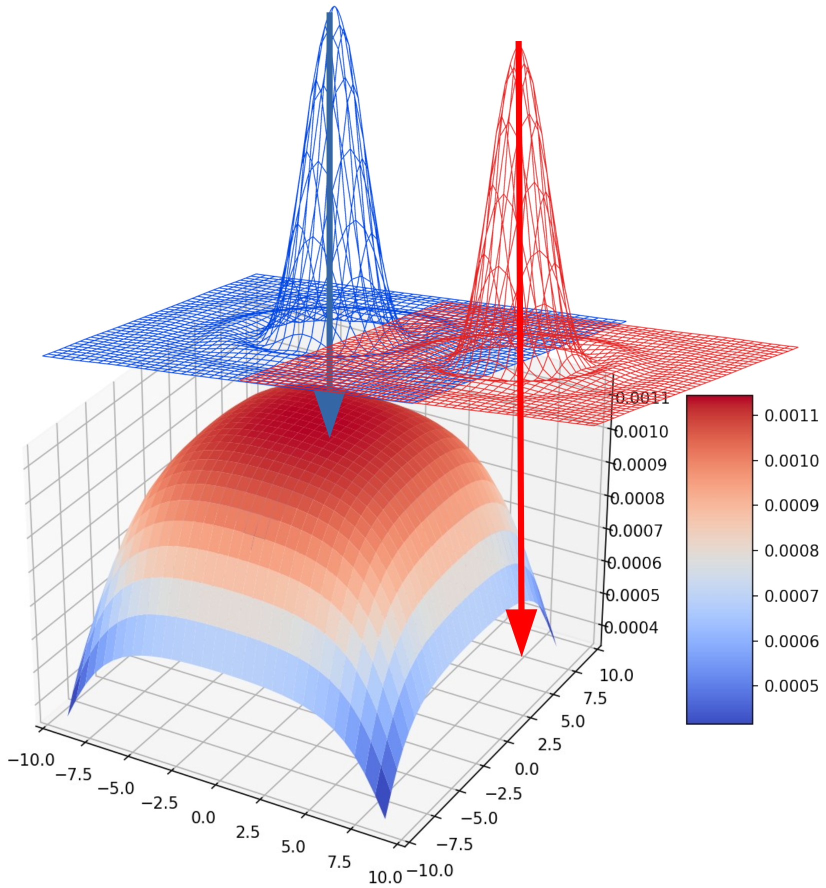
Here are same examples of **flat field** (used to fix this effect) for NIRCam.

The pictures show the effect of
- Different readout groups
- Bad pixels
- Thinner layers of epoxy

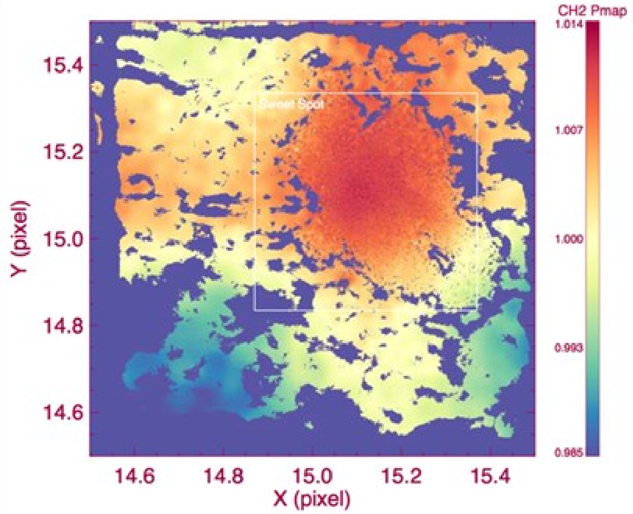


credits

Pixel sensitivity map

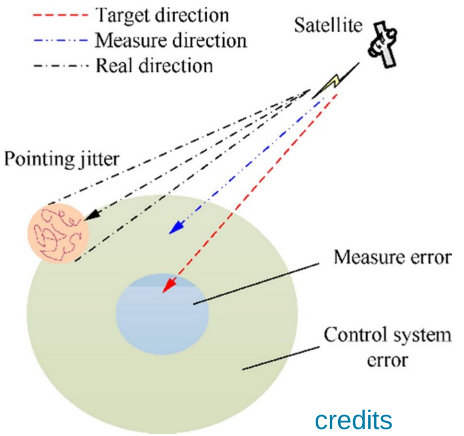


Remember?
Pixel response is not the same in all the surface



Jitter noise

- Variations on the pointed position in the sky **move the light around** the detector
- Light can fall out of the pixel border
- Changed in the detector responsivity
- It is a **colored-noise**: correlated!



credits

The Astronomical Journal, 159:109 (9pp), 2020 March — Morvan et al.

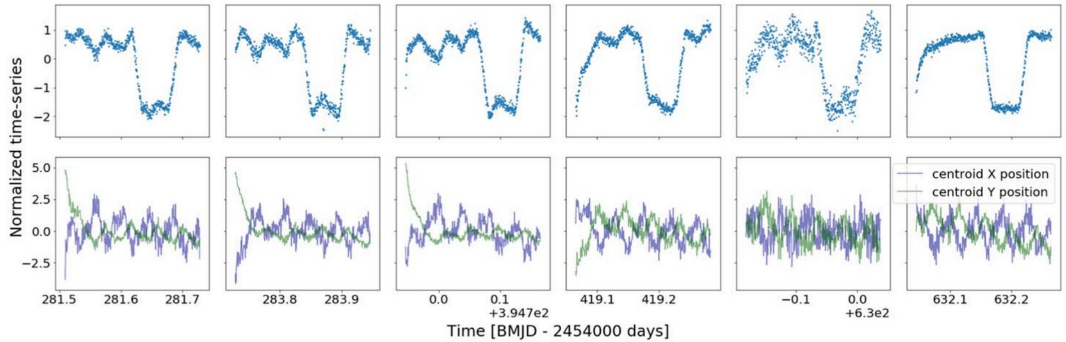
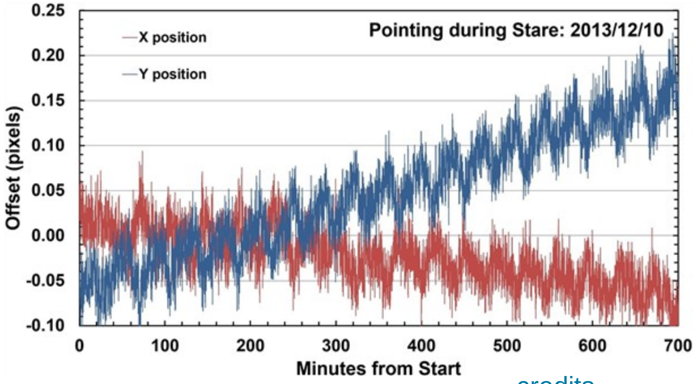


Figure 5. Top: six *Spitzer*/IRAC 8μm raw transit light curves of HD 189733b after preprocessing. Bottom: X/Y centroid positions of the PSF.

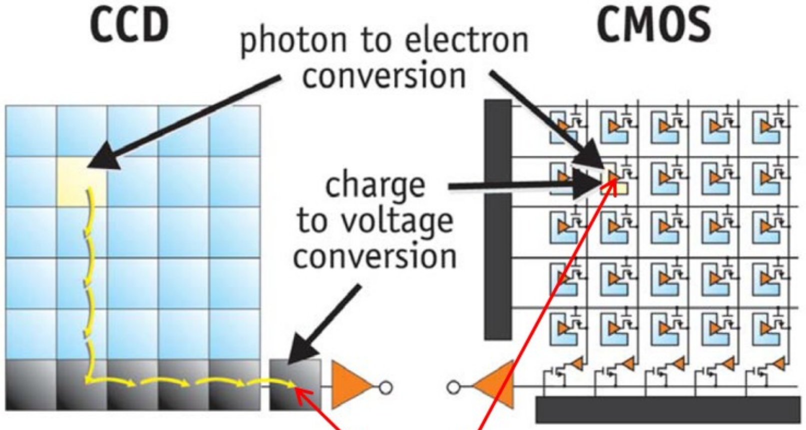


credits

How to read

Detectors can be read at row level or at pixel level, depending on their nature.

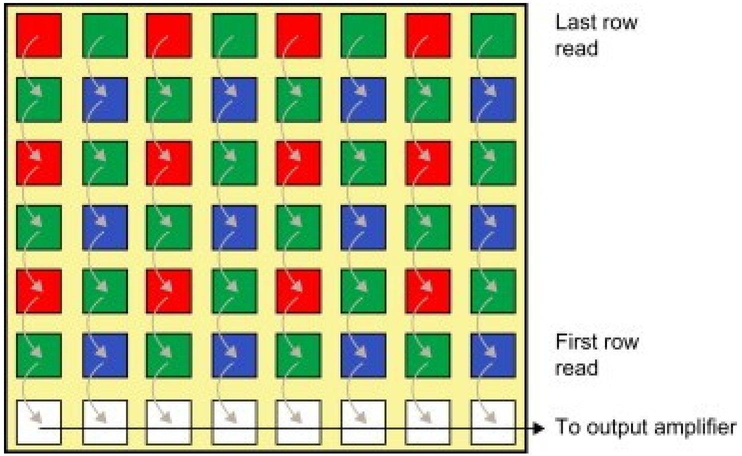
The CCD image sensor shifts one whole row at a time into the readout register. The readout register then shifts one pixel at a time to the output amplifier.



CCDs move photogenerated charge from pixel to pixel and convert it to voltage at an output node. CMOS imagers convert charge to voltage inside each pixel.

credits

Read-out noise generated



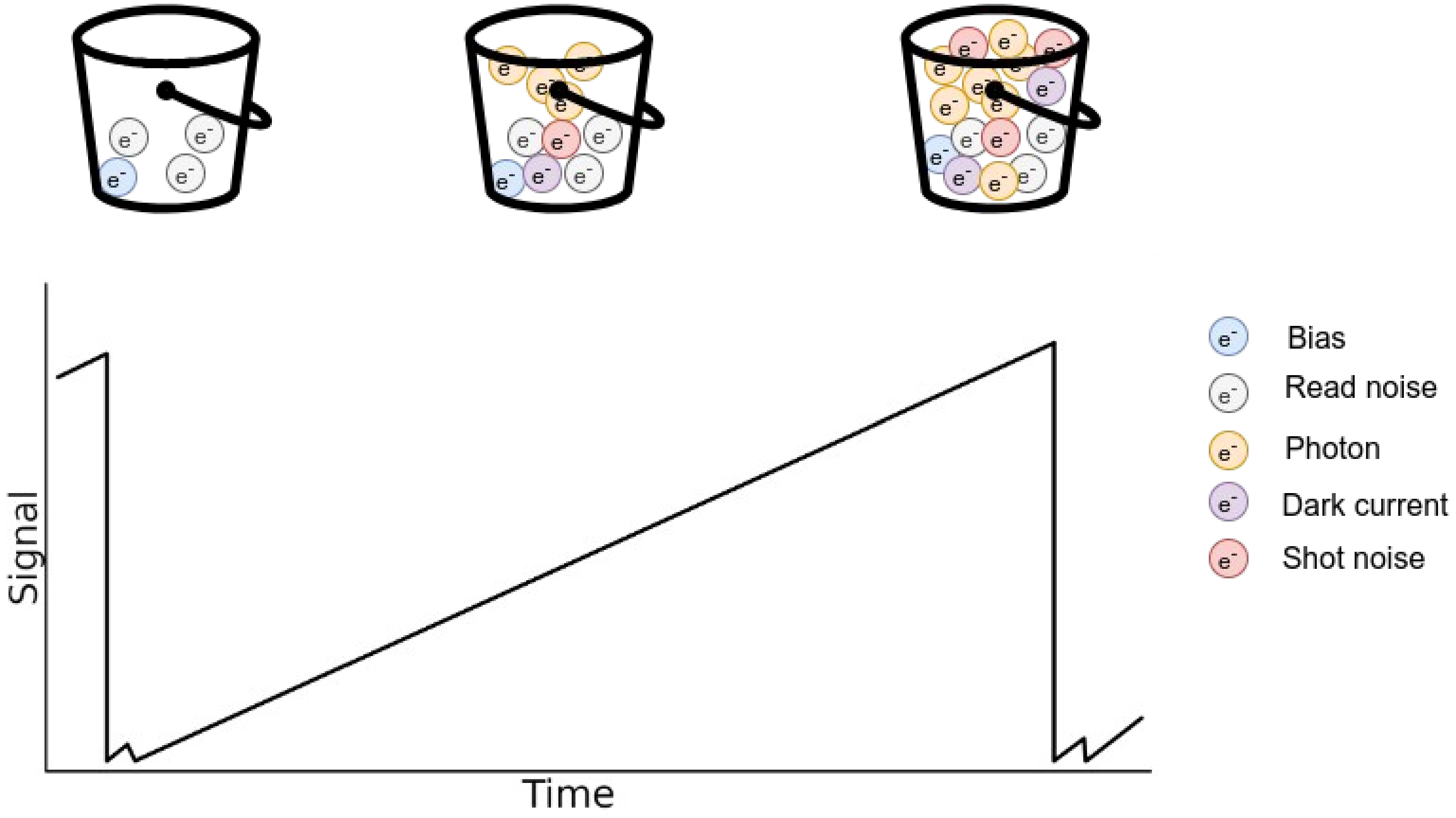
credits

Read out noise

Every time a detector is “read”, some electrons are added to the readings because of the electronics.

This is a Gaussian process (white noise).

Who charges the pixel?



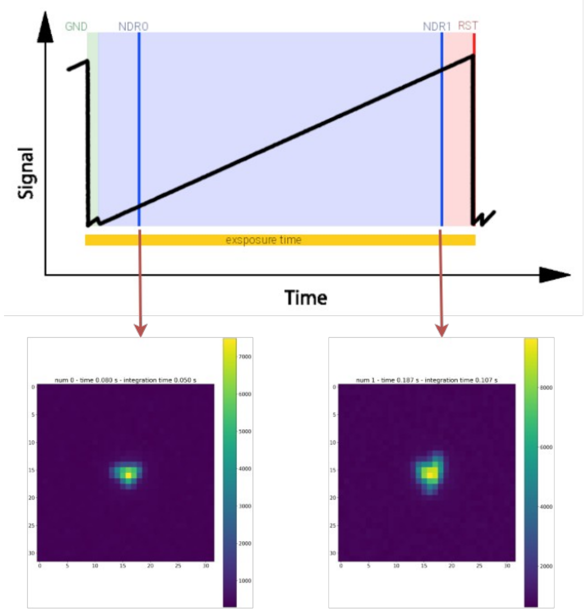
How to sample the ramp

- To extrapolate the flux, you need to **sample the charging ramp**.
- You can use how many **sub-exposures** as you want and combine in how many groups.
- The strategy will **affect your estimated noise**: it will affect photon noise and read noise differently (multiaccum equation)

$$gain_{read} = \frac{12(n-1)}{mn(n+1)}$$

$$gain_{phot} = \frac{6(n^2+1)}{5n(n+1)}(n-1)t_g + \frac{2(m^2-1)(n-1)}{mn(n+1)}t_f$$

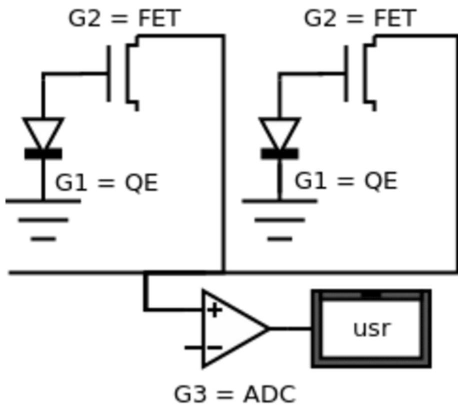
Correlated double sampling (CDS) $\frac{ndr_1 - ndr_0}{\Delta t}$



But our measurements are not in e⁻

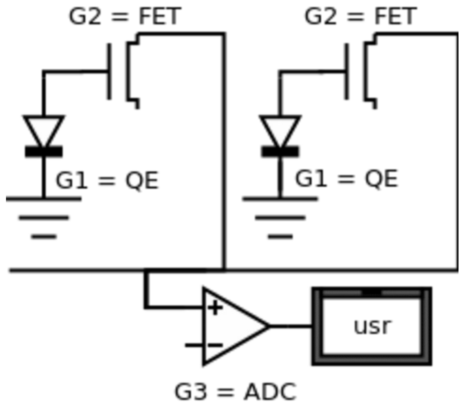
The number of electrons are converted to *adu* (counts) by the **Analog-to-Digital Converter** (ADC).

The ADC is converts a certain number of e- into an integer.



Circuit equivalent for a detector

Circuit equivalent for a detector



G1 is the pixel QE (diode)
G2 is the pixel transistor
G3 is the ADC (common for all pixels)
ɣ is the number of incoming photons
S is the photon counts

$$S = G_1 G_2 G_3 \gamma$$

$$var(S) = var(G_1 G_2 G_3 \gamma) = G_2^2 G_3^2 var(G_1 \gamma) = G_2^2 G_3^2 G_1 \gamma = G_2 G_3 S$$

Where Poisson noise is applied only to the diode.

The variance on the counts is modified from the Poisson noise by a factor $A = G_2 G_3$

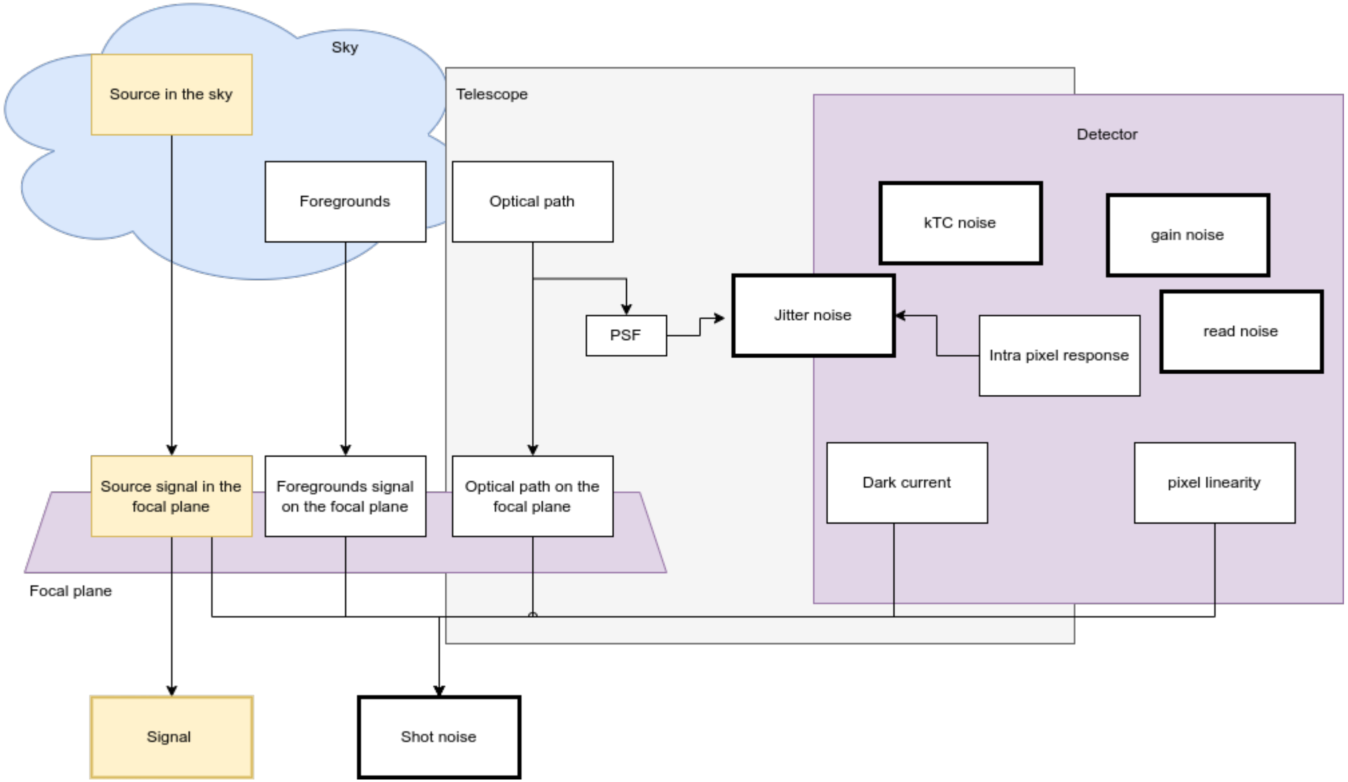
The amplifiers are not perfectly stable over time: they inject time correlate noise!

It is impossible in real life to forge two identical amplifiers!

Noise map (or maze)

Different effects contribute to the final measurement and to its uncertainty:

- Signal from the sky
- Signal from the telescope
- Optical transmission
- PSF
- Detector efficiency
- Observational stability
- Read-out strategy
- Instrumental condition
- Etc.



Noise are tailored for the target

- The estimated noise are made of different components.
  Only one of them depends directly from the target incoming light.
- You should never re-scale the noise to adapt for different targets!
- “Photon noise dominated” doesn’t mean “photon noise only”
- Only uncorrelated noise scales with the number of observations

ExoRad 2

The generic point source radiometric simulator.

Given the payload design and a target, it returns observation performance estimates in terms of noise and SNR.

It analyses hundreds of targets in minutes.

Already validated and used for

- Ariel (*ArielRad*, Mugnai et al 2020)
- Excite

It **does not** require a data reduction pipeline to analyse the data.

How to install

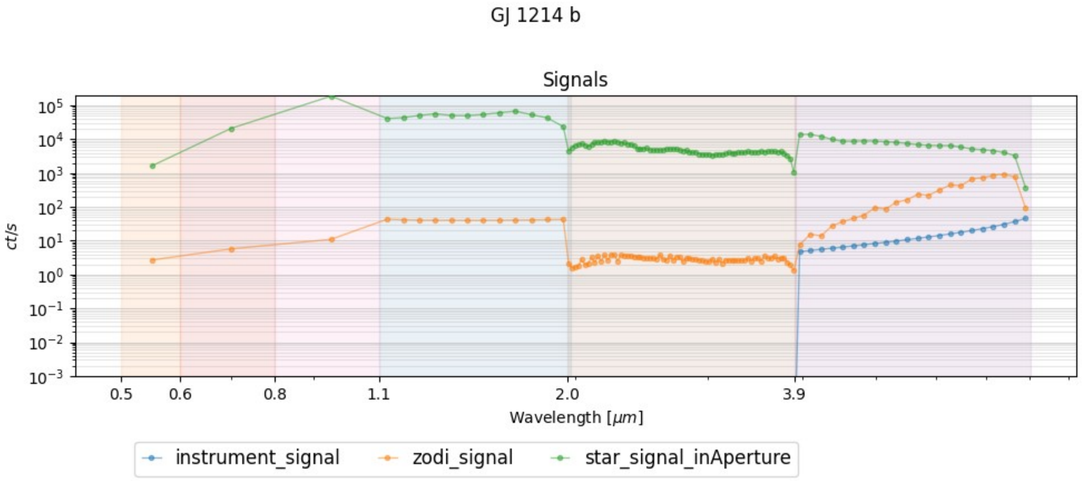
https://github.com/ExObsSim/ExoRad2-public

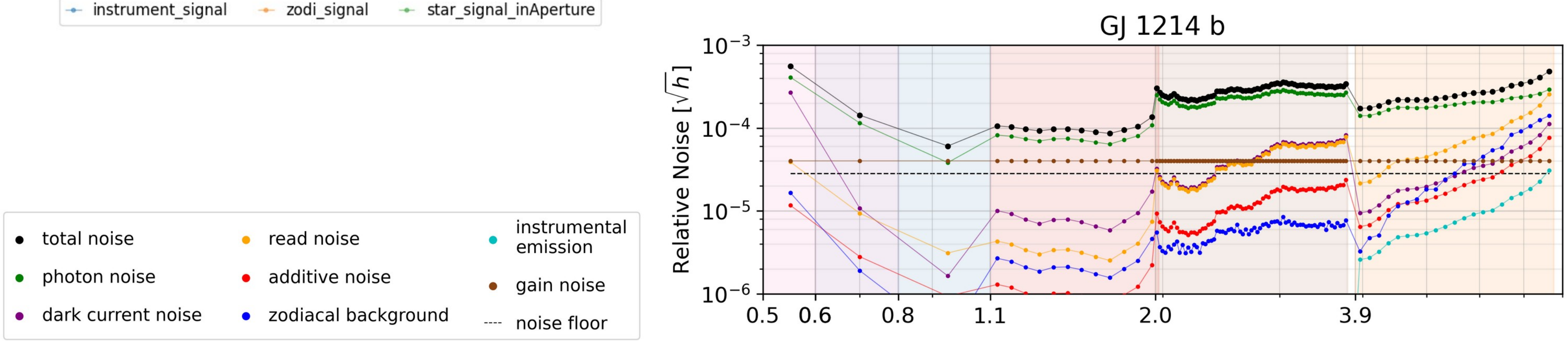
pip install exorad

Documentation

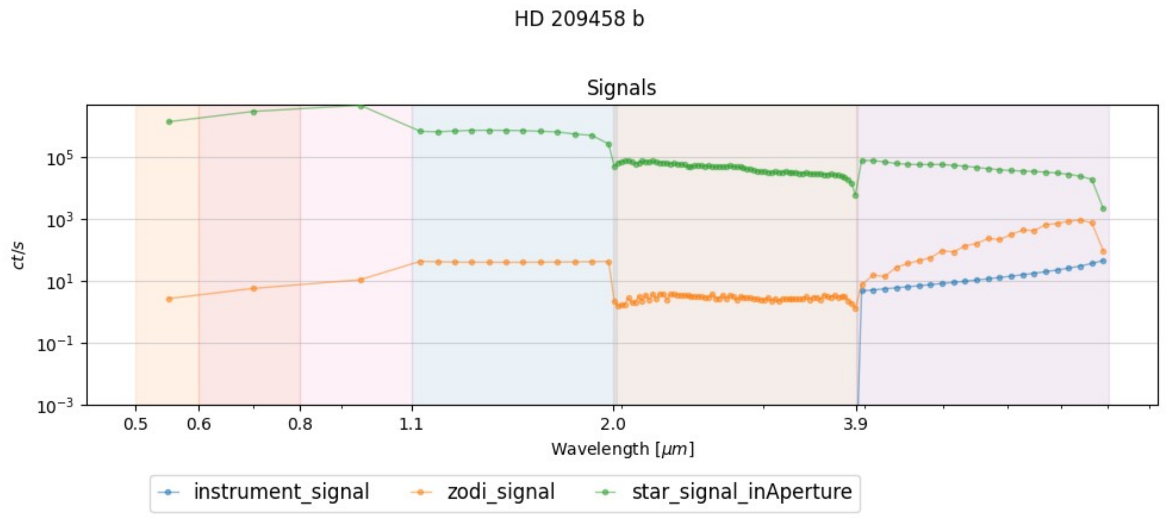
https://exorad2-public.readthedocs.io/en/latest/

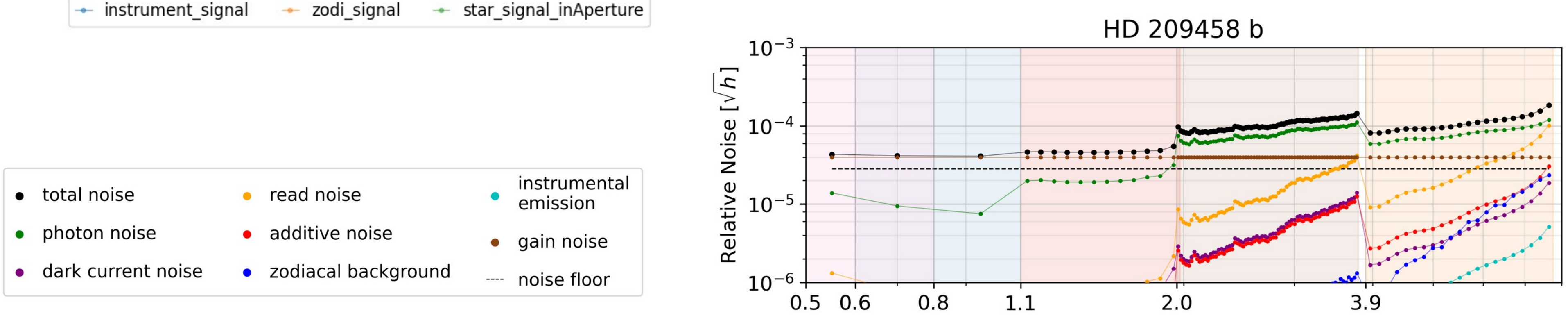
Estimated signal and noise





Estimated signal and noise

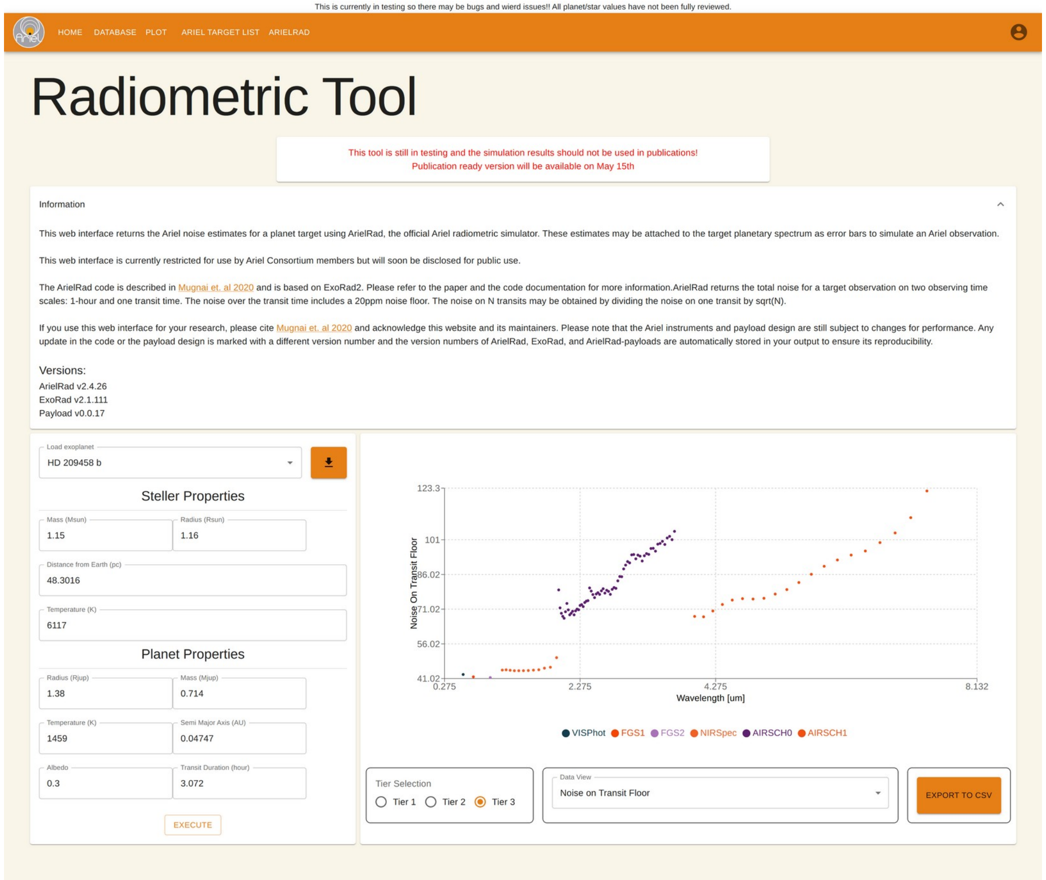




ArielRad online

An online version of ArielRad is available to the consortium members, providing the ETC estimates for the primary science case.

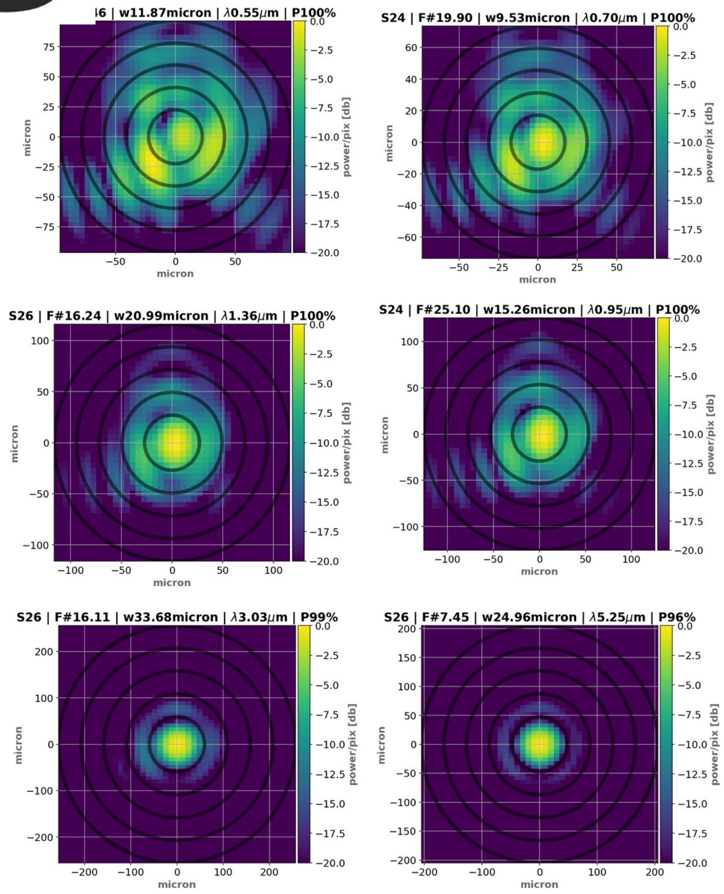
It is hosted on ExoDB.



Among the parameters needed to estimate the max signal in pixel and exposure time is the system PSF.

We use *PAOS*:

- does physical optics;
- accounts from measured aberrations;
- predicts the system PSF.

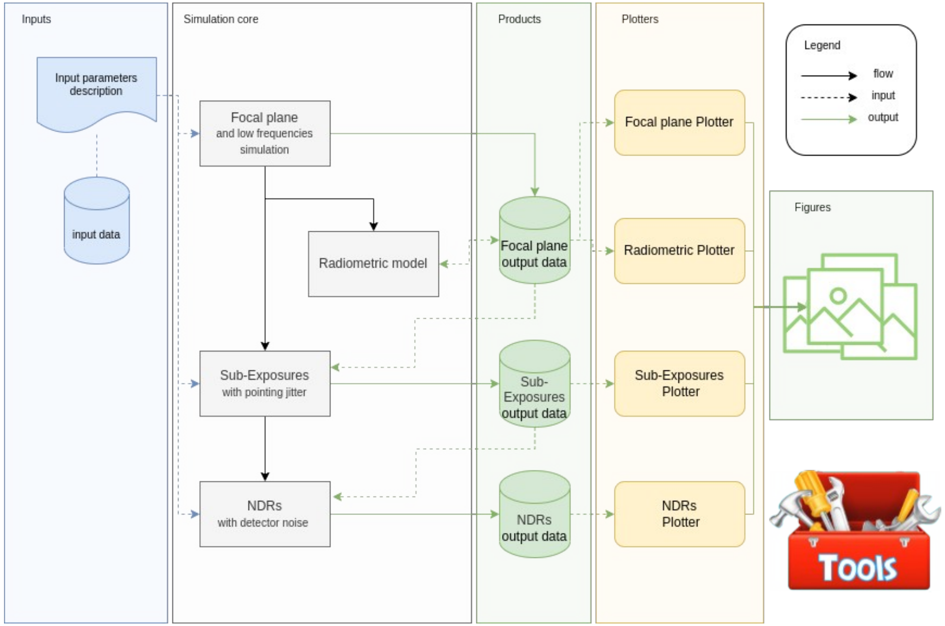


ExoSim 2

The time domain simulator that is
- easy to use than its predecessor
- largely customizable
- completely written in Python
- tested against Python 3.8+,
- follows the object-oriented philosophy.
- fast (~3 min for a 10h simulated observation… on 40 cores)

It comes with
- an installer
- documented examples
- a comprehensive guide

and almost every part of the code can be replaced by a user-defined function, which allows the user to include new functionalities to the simulator



Stats

32.252 python lines

7208 docs lines → (~250 PDF pages)

94% code tested

Use the docs!

Thank you

Lorenzo V. Mugnai
(MugnaiL@cardiff.ac.uk)

Pic. credits Lorenzo V Mugnai